Amendments to the Claims:

The listing of claims will replace all prior versions, and listings, of claims in the application:

1. (currently amended) A method for providing dynamic interaction between a pair of application programs by an interface module of a terminal, the pair of applications including a requestor application desiring access to a target application, the method comprising the steps of:

registering access information of the target application, the access information including published access information made available in a data structure for retrieval by the interface module;

receiving an access request by the interface module from the requestor application, the access request including content corresponding to the published access information of the target application;

obtaining an interface component by using the content to search the data structure, the interface component including an application program interface (API) configured in a language incompatible with the interface module;~~, the interface component further including~~

identifying a plug-in access handler associated with the interface component, the plug-in access handler configured to be specific to the target application and ~~access handler~~ for translating the incompatible language between the API and the interface module; and

employing the interface component by the interface module to satisfy the access request of the requestor application for interaction with the target application.

2. (cancelled)

3. (currently amended) The method according to claim 1 ~~2~~, wherein the incompatible language is that used by a native runtime environment of the terminal.

4. (cancelled)

5. (cancelled)

6. (previously presented) The method according to claim 1 further comprising the step of registering the access handler with the interface module through an extension interface, the published access information of the access handler being added to the data structure.

7. (previously presented) The method according to claim 6 further comprising the step of accessing the target application through the interface module using the access handler to call a corresponding application program interface (API).

8. (previously presented) The method according to claim 7 further comprising the step of employing a search algorithm with the request content for identifying matching ones of the access handlers for use by the interface module.

9. (previously presented) The method according to claim 8, wherein the language used to express the interface module is selected from the group consisting of a structured definition language and a script.

10. (original) The method according to claim 9, wherein the structured definition language is based on XML.

11. (original) The method according to claim 9, wherein the language used to express the script is ECMAscript.

12. (previously presented) The method according to claim 7 comprising a further step of assembling the request content to include selected from the group consisting of a local location and a remote location.

13. (original) The method according to claim 12, wherein the remote location is on another terminal coupled to said terminal through a network, the other terminal having one of the pair of applications for network interaction with the other of the pair of applications.

14. (previously presented) The method according to claim 13, wherein said other terminal is configured as a client of a schema defined service accessible over the network.

15. (previously presented) The method according to claim 7, wherein the data structure is selected from the group consisting of an application profile table and an application API descriptor table.

16. (original) The method according to claim 15, wherein the application profile table includes application profiles of a plurality of target applications.

17. (previously presented) The method according to claim 15, wherein the application API descriptor table includes descriptors selected from the group consisting of API descriptors and access handler descriptors.

18. (previously presented) The method according to claim 15, wherein the data structure includes the access information selected from the group consisting of application URI, application version, application description, and a predefined set of matching API construct pairs.

19. (previously presented) The method according to claim 7 comprising a further step of providing an interface of the interface module selected from the group consisting of an extension interface, a query and registration interface, and an execution interface.

20. (previously presented) The method according to claim 19, wherein the extension interface is configured for dynamically extending a coupling of a new said interface component to the interface module.

21. (currently amended) A terminal for providing dynamic interaction between a pair of application programs in a platform neutral environment provided by the runtime environment of the terminal, the pair of applications including a requestor application desiring access to a target application, the terminal comprising memory and a computer processor configured to implement instructions stored on the memory, the instructions configured to provide:

a data structure for registering access information of the target application, the access information including published access information;

an interface module for providing the platform neutral environment, the interface module configured for receiving an access request from the requestor application, the access request configured to include content corresponding to the published access information of the target application, the published access information of the data structure retrievable by the interface module; and

an interface component coupled to the interface module retrievable by using the content to search the data structure, the interface component including an application program interface (API) configured in a language incompatible with the interface module;~~, the interface component further including~~

a plug-in access handler associated with the interface component, the plug-in access handler configured to be specific to the target application and ~~access handler~~ for translating the incompatible language between the API and the interface module;

wherein employing the interface component by the interface module satisfies the access request of the requestor application in interaction with the target application.

22. (cancelled)

23. (currently amended) The terminal according to claim 21 ~~22~~, wherein the incompatible language is that used by the native runtime environment of the terminal.

24. (cancelled)

25. (cancelled)

26. (currently amended) The terminal according to claim 21 ~~25~~ further comprising an extension interface for registering the access handler with the interface module, the published access information of the access handler being added to the data structure.

27. (previously presented) The terminal according to claim 26 further comprising a corresponding application program interface (API) callable by the access handler for accessing the target application through the interface module.

28. (previously presented) The terminal according to claim 27 further comprising a search algorithm for using the request content to identify matching ones of the access handlers for use by the interface module.

29. (previously presented) The terminal according to claim 28, wherein the language used to express the interface module is selected from the group consisting of a structured definition language and a script.

30. (original) The terminal according to claim 29, wherein the structured definition language is based on XML.

31. (original) The terminal according to claim 29, wherein the language used to express the script is ECMAscript.

32. (previously presented) The terminal according to claim 27, wherein the request content is configured to include selected from the group consisting of a local location and a remote location.

33. (original) The terminal according to claim 32, wherein the remote location is on another terminal coupled to said terminal through a network, the other terminal having one of the pair of applications for network interaction with the other of the pair of applications.

34. (previously presented) The terminal according to claim 33, wherein said other terminal is configured as a client of a schema defined service accessible over the network.

35. (previously presented) The terminal according to claim 27, wherein the data structure is selected from the group consisting of an application profile table and an application API descriptor table.

36. (original) The terminal according to claim 35, wherein the application profile table includes application profiles of a plurality of target applications.

37. (previously presented) The terminal according to claim 35, wherein the application API descriptor table includes descriptors selected from the group consisting of API descriptors and access handler descriptors.

38. (previously presented) The terminal according to claim 35, wherein the data structure includes the access information selected from the group consisting of application URI, application version, application description, and a predefined set of matching API construct pairs.

39. (currently amended) The terminal according to claim 21 ~~22~~ further comprising an interface of the interface module selected from the group consisting of an extension interface, a query and registration interface, and an execution interface.

40. (original) The terminal according to claim 39, wherein the extension interface is configured for dynamically extending a coupling of a new said interface component to the interface module.

41. (original) The terminal according to claim 39, wherein the query and registration interface is configured for publishing the access information related to the interface component.

42. (currently amended) A memory comprising instructions for providing dynamic interaction between a pair of application programs in a platform neutral environment provided by a runtime environment of a terminal, the pair of applications including a requestor application desiring access to a target application, the memory comprising instructions for execution by a processor to implement:

a data structure module for registering access information of the target application, the access information including published access information;

an interface module coupled to the data structure module for providing the platform neutral environment, the interface module configured for receiving an access request from the requestor application, the access request configured to include content corresponding to the published access information of the target application, the published access information of the data structure module retrievable by the interface module; and

an interface component module coupled to the interface module, the interface module configured for containing an interface element retrievable by using the request content to search the data structure module, the interface component including an application program interface (API) configured in a language incompatible with the interface module;~~, the interface component further including~~

a plug-in access handler associated with the interface component, the plug-in access handler configured to be specific to the target application and ~~access handler~~ for translating the incompatible language between the API and the interface module;

wherein employing the interface component by the interface module satisfies the access request of the requestor application in interaction with the target application.

43. (currently amended) A method for providing dynamic interaction between a pair of application programs by an interface module of a terminal, the pair of applications including a requestor application desiring access to a target application, the method comprising the steps of:

registering access information of the target application, the access information including published access information made available in a data structure for retrieval by the interface module;

receiving an access request by the interface module from the requestor application, the access request including keywords associated with the published access information of the target application;

obtaining an interface component by using the keywords to search the data structure, the interface component including an application program interface (API) configured in a language incompatible with the interface module;~~,~~ ~~the interface component further including~~

identifying a plug-in access handler associated with the interface component, the plug-in access handler configured to be specific to the target application and ~~access handler~~ for translating the incompatible language between the API and the interface module; and

employing the interface component by the interface module to satisfy the access request of the requestor application for interaction with the target application.